IDENTITY MANAGEMENT SYSTEM

AUTHENTICATION: CORE COMPONENTS

UMD Directory
The Utah Master Directory (UMD) includes two sets of users: State employees and the public. Under the control the Department of Human Resources, employee records can be added, deleted, or modified and instantly updated in the UMD database. Public accounts are created and maintained through the Credential Collector Web form (see Credential Collector below). UMD also synchronizes user account data with the LAN Directory for automatic provisioning. Details of this process can be found in Appendix A.

Access to UMD data, via LDAP, is strictly controlled. Recommended access is through SiteMinder or the AppProfile system (see AppProfile below).

Site Minder
SiteMinder is a Netegrity product that provides highly secure authentication and authorization of Web resources. An encrypted cookie that contains SiteMinder session information is stored on the browser and accessed by each protected utah.gov domain upon visitation. Once a user has authenticated using UMD the use of this encrypted session information provides a single sign on environment.

ID and password make up the currently supported authentication schemas. The password is stored in the directory as a public/private key pair and is not retrievable. SiteMinder must authorize every URL in the protected realm. Authorization or access is allowed based upon information in the user's UMD account. For example, an account exists in the "employee" container, or, an account has the agency code set to "100".

AUTHENTICATION AND DIRECTORY ACCESS LAYER

Credential Collector.
The credential collector is a J2EE Web application created by ITS. Its functions are to:

- provide a standard login ID to the SiteMinder system (this is the directory DN);
- provide a standard password restoration method for forgotten passwords;
- provide a means to create public user accounts;
- provide for email address verification (for creating "trust" in the account; and,
- provide account maintenance functions for both public and employee accounts.

Users may enter their alias/userID, email address, or employee ID number to login. The alias/userID can be set to anything the user desires and may be changed by the user as long as the value remains unique. The email address can be changed by public users and triggers an email address verification process. For details on the login process, see Appendix B.

AppProfile System.
The application profile system is a client/server architecture written in Java. It provides flexible, secure, access to the UMD. Similar to a database, multiple AppProfile servers provide the data access functionality, complete with failover, while the AppProfile Client provides the means for programs to access the server's functionality

Features include:

- The ability to extend the UMD schema for the addition of custom attributes.
- A variety of field types: Binary, Selection, Option, Text, Existing-Text, Encrypted-Text, Date, and Number.
- Access controlled by agencies
- Controlled access based upon a "scope", limiting the view of accounts.
- The ability to group profile attributes by type.
- The ability to give values to groups of user accounts based upon a priority level.
- The ability to search for accounts in a profile using specified attribute values.
- The ability to search the general directory of accounts within an authorized scope.
- A caching system for storing retrieved data for quick reference.

APPLICATION INTEGRATION AND ADMINISTRATION

ApAdmin
ApAdmin is a J2EE Web application that provides a general-purpose implementation of the AppProfile Client. All ApAdmin functions are directly available to programmers using the AppProfile Client, removing the necessity of learning the complexities of schema administration. In addition, ApAdmin may fulfill the requirements of application administration, eliminating the need to create a custom administration interface.

JAAS Providers to Application Servers
Java Authentication and Authorization Service (JAAS) is a standard API (Application Program Interface) that application developers can use when creating applications. While developing the application, a developer may use a file based JAAS provider. Then, when ready for deployment to a SiteMinder/UMD protected server, the JAAS roles are simply mapped to roles stored in UMD.

Providers have been created for WebSphere, SUN AppServer 7, and Tomcat using information from the AppProfile system to provide role membership information to JAAS.

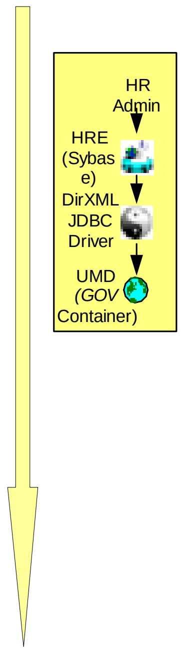
OTHER FUTURE SERVICES

Other services that might be developed in this space include:

- WebService to deliver UMD/AppProfile information to non-Java environments.
- JDBC Connector for AppProfile information.
- Other task specific programs for extracting information from UMD for Billing, etc.

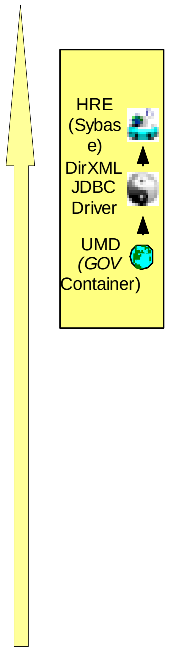
Appendix A

HR Enterprise to UMD



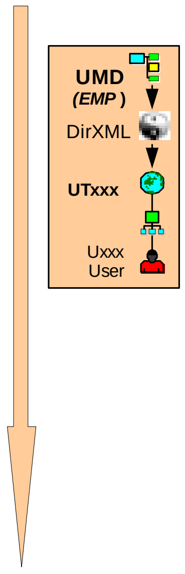
CN
workforceID
Department
Full Name
Generational Qualifier
Given Name
Initials
Internet EMail Address
Manager
Physical Delivery Office
Postal Code
Postal Office Box
Private Key
Public Key
S - State
SA - Street Address
Surname
Telephone Number
Title
UTIW-AgencyNumber
UTIW-BldgID
UTIW-DistList
UTIW-EmpEmail
UTIW-InetMailDomain
UTIW-InsurEligible
UTIW-Invisible
UTIW-LeaveEligible
UTIW-LowOrg
UTIW-NWAccess
UTIW-Placement
UTIW-PO
UTIW-PODomain
UTIW-PrimaryResTree
UTIW-PublicEmail
UTIW-Status
UTIW-UserTemplate

UMD to HR Enterprise



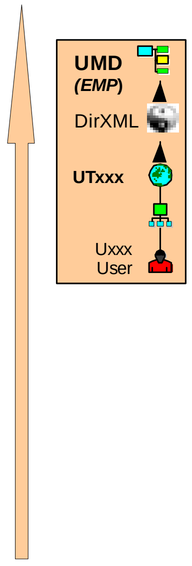
Given Name
UTIW-PublicEmail
Telephone Number
Internet Email Address

UMD to Existing State Directory



Description
Facsimile Number
Full Name
Generational Qualifier
Given Name
Initials
Internet Email Address
Login Disabled
mailstop
mobile number
pager number
Password Required
Physical Delivery Office
Postal Code
Postal Office Box
Private Key
Public Key
S
SA
Surname
Telephone Number
Title
UTIWAgencyNumber
UTIW-EmpEmail
UTIWInetMailDomain
UTIW-Invisible
UTIW-LowOrg
UTIW-NWAccess
UTIW-Placement
UTIW-udotCN
UTIWudotPlacement
UTIW-usorCN
UTIWusorPlacement
UTIW-utcourtsCN
UTIWutcourtsPlacement
UTIW-utstateCN
UTIWutstatePlacement
workforceID

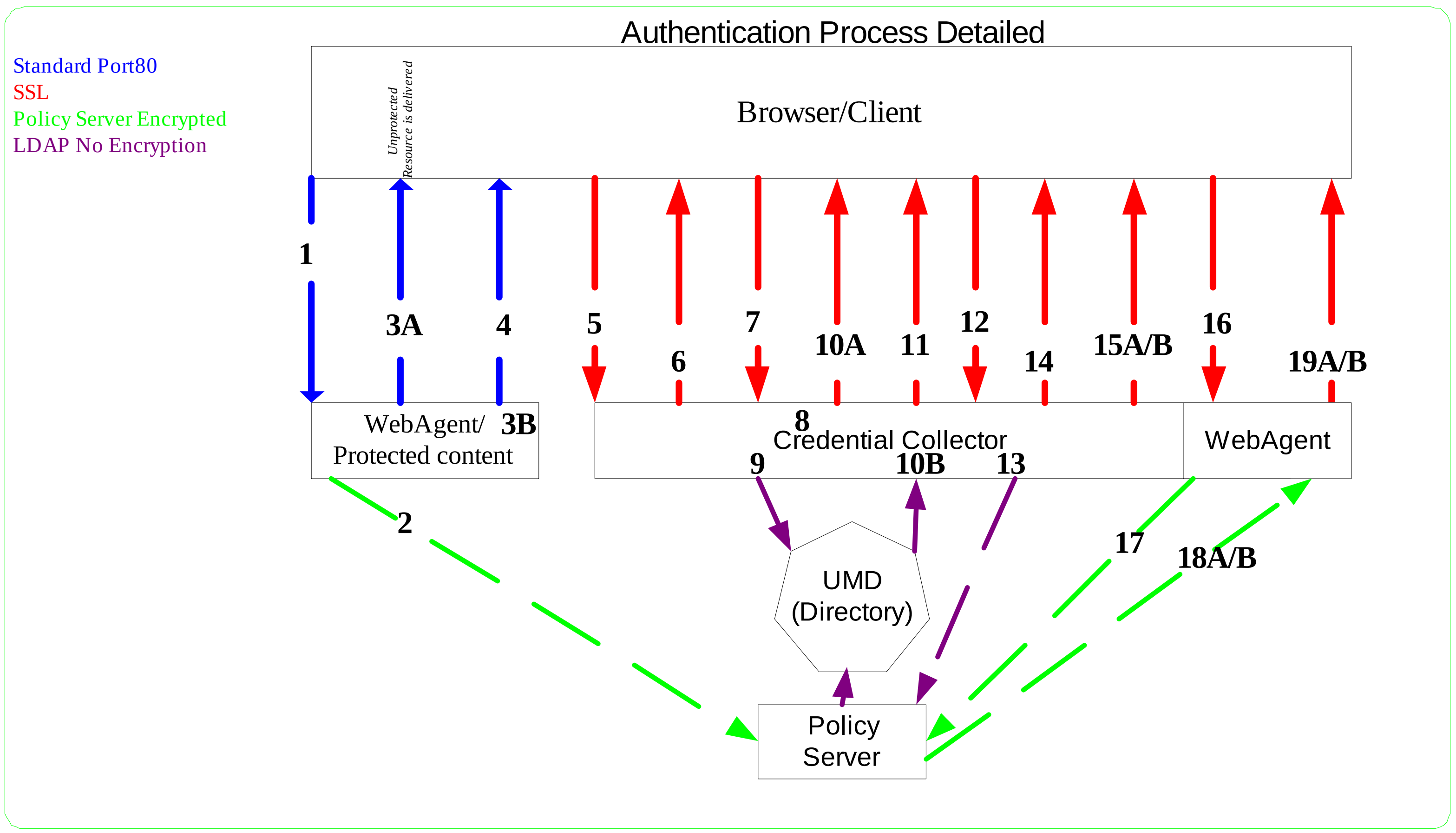
Existing State Directory to UMD



Facsimile Number
Given Name
Internet Email Address
mailstop
mobile number
pager number
Password
Name
Surname
Telephone Number
Title
UTIW-EmpEmail
UTIW-Placement
UTIW-udotCN
UTIWudotPlacement
UTIW-usorCN
UTIWusorPlacement
UTIW-utcourtsCN
UTIWutcourtsPlacement
UTIW-utstateCN
UTIWutstatePlacement
workforceID

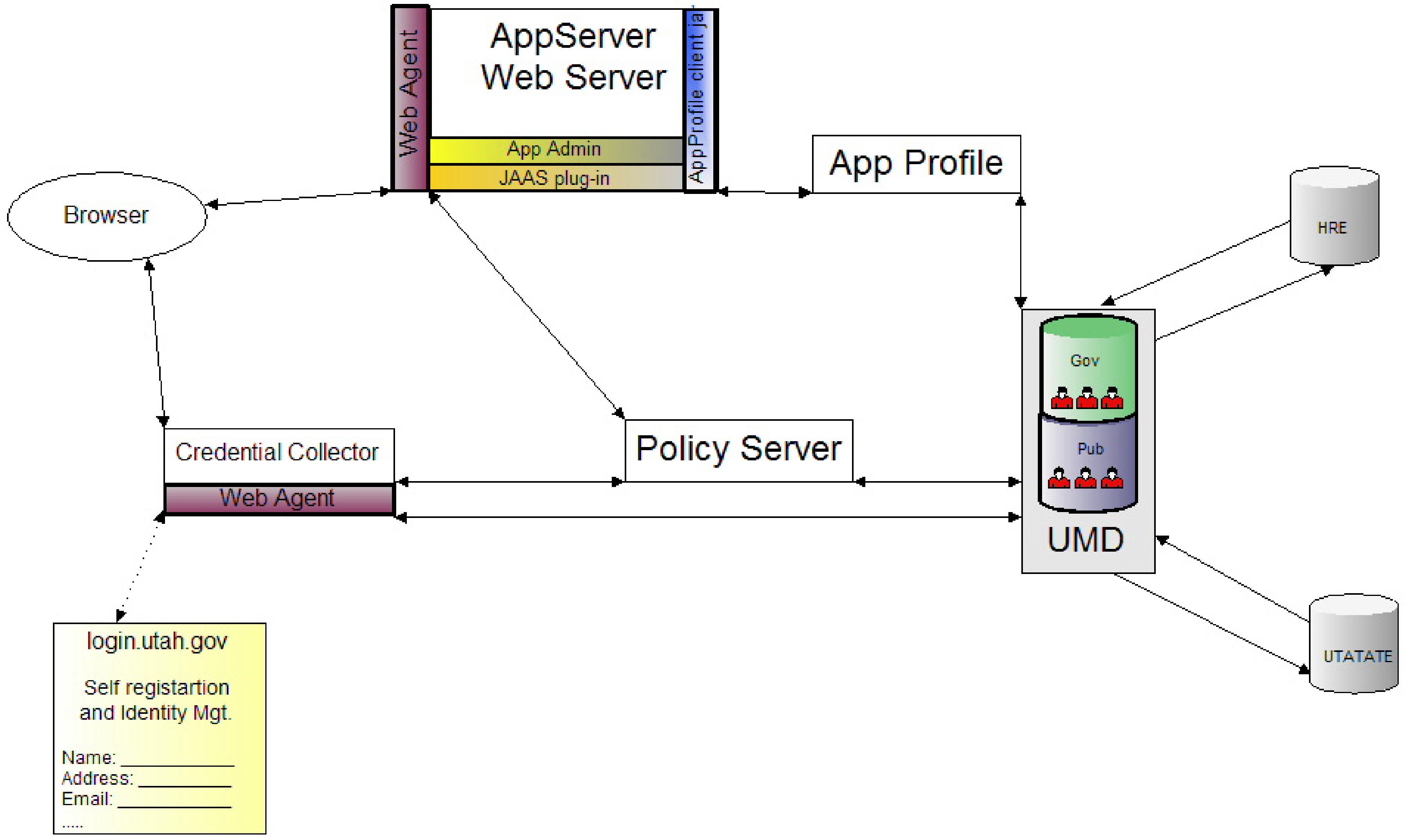
Note: UTxxx = UTSTATE, UDOT or USOR.
PhaseII will include Courts and BOE

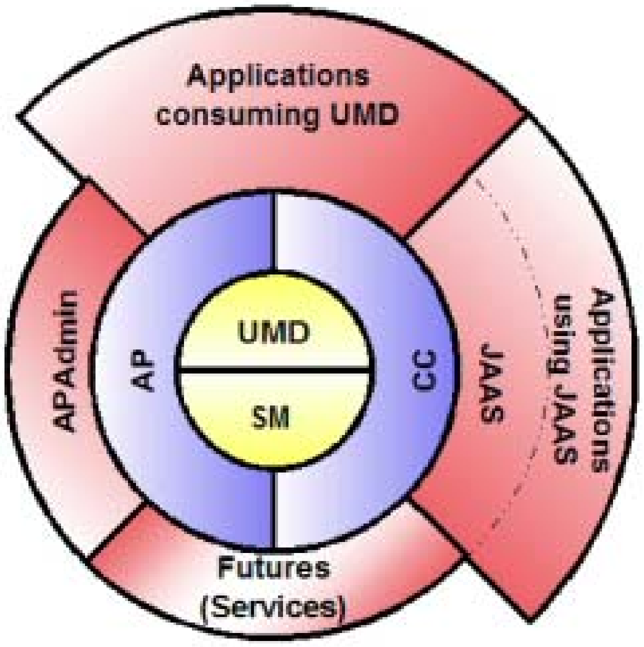
Appendix B



Stepping through the authentication process

Step 1.	Client Browser makes a request to a resource.
Step 2.	WebAgent checks local cache for protection rules, then checks policy server for protection rules
Step 3A.	If resource is not protected web agent is instructed to serve resource.
Step 3B.	If resource is protected webagent is instructed to serve authentication realm
Step 4.	Client receives a redirect to the credential collector as specified in the Authentication realm.
Step 5.	Client processes redirect and request login page from credential collector
Step 6.	Credential collector serves the login page
Step 7.	Client fills out the login form and then posts data.
Step 8.	Post data is messaged by the credential collector.
Step 9.	Credential collector checks clients password.
Step 10A.	If there is a password error client is sent an error page that contains another login form.
Step 10B.	If password checks out, clients full DN is found
Step 11.	Credential collector sends back an auto post form with full DN and password to client
Step 12.	Client processes form and some JavaScript auto submits the form to siteminder login.fcc
Step 13.	Siteminder policy server and webagent approves or denies access based on policies in the rules and protection policies in the policy server
Step 14.	The appropriate page will be sent to client.
Step 15A.	Login Successful Credential collector send a redirect to the desired target.
Step 15B.	Access Denied or any other error page
Step 16.	Success page redirect is processed and client requests resource again.
Step 17.	WebAgent checks sm credentials with local cache and or policy server if necessary.
Step 18A.	Policy server responds to webagent access is granted.
Step 18B.	Policy server responds to webagent access is denied based on rules and policy information in policy store an error message will be sent back to the client.
Step 19A.	WebAgent is instructed to serve resource and serves the resource to the client
Step 19B.	WebAgent is instructed to serve an error page with another login scr


Web Agent
AppServer
Web Server
AppProfile client ja
App Admin
JAAS plug-in
App Profile
Browser
HRE
Gov
Pub
UMD
Policy Server
Credential Collector
Web Agent
UTATATE
login.utah.gov
Self registartion
and Identity Mgt.
Name: ___________
Address: __________
Email: ___________
.....



Core Layer

UMD	=	Utah Master Directory
SM	=	Site Minder

Authentication and Directory Access Layer

AP	=	AppProfile
CC	=	Credential Collector

Application Integration and Administration

APAdmin	=	Application Administration
JAAS	=	Java Authentication and Authorization Service
Applications	=	Applications consuming one or more of the other layers
Futures	=	Services could include PHP, Perl, custom